NUID: 001738408
Name: Siyu Qiu
Course: Foundations of Artificial Intelligence

#Assignment 1

In this Project, I implement 3 algorithms to get the best move for Pacman.

All the controllers are included in the package pacman.controllers.Siyu_Qiu

I created a PacManNode class to keep track of the depth of each Game state. For BFS, I tried each possible move to find which path will get the highest score by Breadth First Search.
BFS will search for the highest score level by level, and going to the next depth after we have done with all the nodes in the current level, and the searching will be terminated at maxDepth and return the current score.

For DFS, I also find the path which has the highest score by Depth First Search.
DFS will search for the highest score by going to the max depth for each node, and will turn to next node after we got the score in maxDepth for this node. The searching will also terminated at maxDepth in each loop.

For Iterative Deepening Search, I get the highest score in each depth and return the highest score among all depth, searching in each depth is handled by Depth First Search.

When we found multiple best moves, in this case, because we have more pills in the upper side of map, going up has bigger possibility to be the first choice for us, so simply choose the first move is fine. However, this may cause Pacman stops when we already at the top of the map. Thus, in most cases, we may want to choose best move randomly from those who got the bestScore. So I also added codes to choose directions randomly. You can simply uncomment them and run the program.

For example:

```
//              else if(tempHighscore==highScore){
////            Add this statement to choose move from the list of
bestMoves
//                    bestMoves.add(m);
//                    }
….
…
//         //Only choose when we found multiple bestMoves
//         if(bestMoves.size()>0)
//              bestMove=bestMoves.get(rnd.nextInt(bestMoves.size()));
//
```

To compile the program, you just need to *add*

```
exec.runGameTimed(<Controller’s name>,new StarterGhosts(),visual);
```

in Executor.(Assume we use StarterGhosts for ghosts)